Jan 23rd

Thursday, January 23, 2014 10:03 AM

Binary Executable: is a static file --------> executed -----> goes to a process: now has a state

HOW PROCESSES RUN: and keeping track of their states

States:

1. Start:
    a. Beginning of the program
2. Ready:
    a. A state where the process is ready to do something
        i. 0 or more process, waiting for CPU time
3. Run State:
    a. 1 process per core in a run state
    b. From here it can more to any state
4. Wait I/O
    a. They are in the state when waiting for something else to happen
    b. After they are done waiting they most likely move back to ready queue
    c. Can have 0 or more processes
5. Terminate
    a. The end of the process
    b. Typically won't return back to another state
    c. 0 or more processes

Process Control Block (PCB): this contains a large amount of information and how the OS keeps track of what is happening with processes

Unique to Each process (try to keep it in its own core for cache reasons: called process affinity)
Registers
State
Scheduling
Memory
I / O

Threads are similar to (PCB)

Unique to Each Thread
Registers
States
Scheduling

Shared by Threads
Memory
i/O

All of these things are controlled by the scheduler:
The scheduler is very important program of the OS that decides which processes go where.

Two Types

1. Long term scheduler
    a. A separate program that checks to see if the right number of processes are running on the system
    b. Determines based on resources available
2. Short term scheduler
    a. Maximizes the CPU utilizations

Context Switch

1. Switches between process
2. Switches between protected and user mode
3. Manages quite a bit of the state of the registers

The Life Cycle of a Process

- Every process starts with a parent
    - This is a system call that creates command and creates a child: Linux this is fork()
        - This creates a new PCB structure
        - Copies the parent PCB into the child PCB
        - Also giving a Change Process Identifier (PID)
        - Then places the child onto the ready queue
            - Ready Queue
                - Parent
                - Child

This is chosen by the operating system scheduler

    - Execute a binary program with EXEC*
        - Man exec
        - This overwrites the memory of the child process with what you specify by the exec command

See Code

Man = manual this is where you can get any function call
Pid = fork

First thing to do is to check for the error code = -1
Next check for PID = 0, this means that it was successful and it